On the importance of similarity measures for planning to learn

Hendrik Blockeel [1,2] and **Hossein Rahmani** [2] and **Tijn Witsenburg** [2]

Abstract. Data analysis is a complex process that consists of finding a suitable data representation, a suitable machine learning method, and using a suitable evaluation metric (one that reflects what the user is really interested in). All these choices are crucial from the "planning to learn" perspective, and none are trivial. In this paper we focus on the first of these three, the input space representation. Sometimes this problem is posed as "defining the right features", but in those cases where we have non-standard data, for instance, for relational or graph data, the data representation problem does not map easily on feature construction. In some sense, it is easier to see it as a problem of constructing a suitable distance metric, similarity metric, or kernel. In this paper we discuss this view in some more detail. We next illustrate it by looking at input data represented as annotated graphs, and defining a few similarity measures in this context. We illustrate the importance of the choice of distance measure with an experiment on the Cora dataset.

1 INTRODUCTION

It is generally agreed that the Knowledge Discovery process comprises much more than the data mining step alone [1]. The user first needs to understand the knowledge discovery problem, the goals of the knowledge discovery task, and the data that will be used as input. Based on this insight, more technical choices need to be made regarding:

- The data representation, or input space representation. This concerns how the input data for the learner are represented. We prefer the term "input space representation" here because "data representation" may suggest that this question relates to how single data elements are represented. In fact, as we will argue further on, the question is not so much how to represent individual data elements, but how to represent the data set as a whole, or how to represent the relative positioning of the elements with respect to each other.
- The machine learning methods to be used. Obviously, different tasks call for different machine learning methods. But even among methods that fulfill the same task (for instance, classification), each method has its particular bias [3]. This bias, however, is only defined relative to the concrete input space being used. In a concrete application context, the bias of the machine learning process is determined by both the input representation and the learning method being used.
- The evaluation metrics: these are directly related to the goals of the knowledge discovery process. They are in a sense a formal translation of the user's success criteria to the context of the pattern language used to describe the output of the learning system.

[1] Department of Computer Science, Katholieke Universiteit Leuven
[2] Leiden Institute of Advanced Computer Science, Universiteit Leiden

In this paper, we investigate in some more detail the data representation question. We focus on the particular case of learning from annotated graphs; these are graphs where nodes, edges, or whole (sub)graphs can have additional information attached to them. We focus more specifically on the case of graphs with annotated nodes, and discuss different ways in which similarity measures can be defined for those graphs. We present experimental results showing that the choice of the distance measure indeed has a significant influence on the results of the classification process, which suggests that this issue may deserve more attention.

2 THE ROLE OF SIMILARITY MEASURES IN LEARNING

Clustering processes typically depend on a notion of similarity (sometimes expressed using a distance metric) between the elements to be clustered. Also classification processes rely on such a notion, sometimes explicitly, sometimes implicitly. In **instance-based learning**, such as k-nearest-neighbor methods, the reliance on a similarity measure is explicit; more specifically, these methods rely on the assumption that instances that are similar tend to belong to the same class. (In the regression context, the corresponding assumption is that the function to be learned is continuous, which is often a reasonable assumption.) Also **kernel-based methods** rely on the notion of similarity, since a kernel function can be interpreted as indicating the similarity between two points. But even a **decision tree learner**, for instance, which is normally not considered a similarity-based method, can be interpreted in those terms. A decision tree essentially expresses the hypothesis that whenever two instances have the same values for a limited number of attributes (so that they end up in the same leaf), they are similar enough to belong to the same class. Learning a decision tree corresponds to identifying what the relevant combinations of attributes are, on the basis of which similarity can be decided; in other words, it corresponds to *learning a suitable similarity measure*. A similar claim can be made for rule learners.

In this sense, it is clear that the bias of a learning method is strongly connected to the notion of a similarity measure in the input space. From the point of view of planning to learn, we believe this is an important observation. It is generally accepted that bringing the input data into the right format (defining the right features, for instance), is an important step on which the performance of a learner will crucially depend. But the notion of similarity is equally important: whereas an incorrect representation may make it impossible for a learner to use a correct similarity measure (e.g., for a decision tree, not having the right features makes it impossible for the tree to express the correct combinations of features), having a correct repre-

sentation does not guarantee that the learner will be able to use the right similarity measure.

Rather than stating that an input format is suitable for learning if it contains the right features, which corresponds to stating that the input space has the right dimensions, it may be more accurate to state that the input space is right if individual elements are embedded into a metric space in such a way that elements that are more likely to have the same class are close to each other.

This view is quite similar to the view taken in kernel-based methods, where the kernel implicitly defines such an embedding into a metric space. The point that we are making here, is that, from the point of meta-learning or planning to learn, it may be useful to pay more attention to construction the right similarity measure, rather than simply constructing the right features.

Example 1 *To make this a bit more concrete: suppose we want to learn the function A XOR B. It is not possible to state whether a set of features is the right one, without considering it together with the learner that will be used; conversely, it is not possible to state whether a learner will be suitable without considering the input features. For a decision tree learner, an input space with features* $\{A,B\}$ *allows the learner to construct a correct decision tree, but for a single-layer perceptron the same input space is not sufficient. If we focus on the notion of similarity, we can say that any embedding of the data into an input space that considers two points to be similar if their A XOR B value is the same, is a good embedding.*

The notion of similarity can thus replace the notion of "combination of input space and learner", and offers in a sense a more principled description of that combination. Stated in yet another way, there is no need to consider different learners; a single instance-based learner, making use of the right similarity measure, can mimic the learning behavior of any other learning method.

3 ANNOTATED GRAPHS

We now introduce the concept of annotated graphs; these will form the basis for an illustration of the points we have been making above.

With an "annotated graph" we mean a graph structure where nodes, edges, (sub)graphs can be annotated with additional information. That is, an annotated graph is a structure (V, E, λ) with V the node set, $E \in V \times V$ a set of edges, and $\lambda : 2^V \rightarrow \mathcal{A}$ a function that assigns to any subset S of V an "annotation" $a \in \mathcal{A}$. The space of possible annotations is left open; it can be a set of symbols from, or strings over, a finite alphabet; the set of reals; an n-dimensional Euclidean space; a powerset of one of the sets just mentioned; etc. When S is a single node or a pair of two nodes (i.e., an edge), we say that the annotation is attached to the node or edge.

In the remainder of this text we will consider the more specific case of graphs where only nodes are annotated. The annotations will typically be treated as vectors or subsets from a predefined set (note that such a subset can always be represented as a boolean vector with one component per possible element, which has the value true if and only if the element is in the subset).

Note that if the edges in this graph structure are ignored, the dataset is reduced to a set of nodes V with each node annotated with a vector and each node standing on its own; since the nodes carry no information except for the vector attached to them, we essentially obtain a standard attribute-value dataset.

Regarding the learning task, we focus on the case where single nodes are the data elements that have to be classified or clustered. *Classification* or *clustering* can be seen as completing the annotation of a node by filling in the value for one particular component of the annotation, the so-called target variable, which indicates the class or cluster the element belongs to.

4 SIMILARITY MEASURES FOR NODES IN ANNOTATED GRAPHS

The discussion in Section 2 is particularly relevant in the context of learning from annotated graphs. The information contained in these graphs is not easily embedded in a finite-dimensional space (i.e., it is not possible to define, in general, a finite set of features that represents all the information in the graph, and which can be used for learning). However, as argued before, what is really relevant is simply the notion of similarity. More specifically, when clustering or classifying nodes, if we can define the right similarity measure for these nodes, we will obtain a good clustering, or accurate classification.

This brings us to the question of defining similarity measures for nodes in an annotated graph. The following discussion is based on earlier work by Witsenburg et al. [5].

When looking at clustering nodes in an annotated graph, we can distinguish "standard clustering" algorithms and "graph clustering" algorithms [5]. In **standard clustering**, items are clustered according to their similarity whilst not taking into account any relational information (that is, all edges in the graph are simply ignored). A distance or similarity function is given that for any pair of items tells us how similar they are. This results in a $N \times N$ similarity matrix. In **graph clustering**, unlabeled graphs are considered, where clustering (or partitioning) the graph typically means finding subgraphs of the graph such that the number of links connecting different subgraphs is as small as possible whilst not taking into account any information about the content of the node. Figure 1 illustrates the difference between these clustering settings.

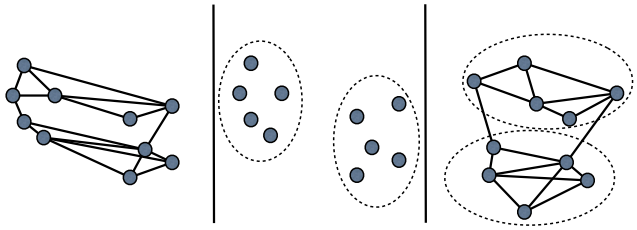


Figure 1. Left: An annotated graph, plotted such that distance between nodes reflects their similarity; Middle: standard clustering ignores the edges and clusters nodes according to their distances; Right: graph clustering methods ignore the actual node positions and typically find clusters such that nodes within clusters are highly interconnected whereas nodes between different clusters are sparsely connected. Standard clustering and graph clustering may yield very different clusters.

Any $N \times N$ matrix can be converted to a graph (with the element A_{ij} representing the weight of the edge between nodes i and j) and vice versa. This raises the question whether, in those cases where both node content and graph structure are available (such as the Web), one could find a clustering method that combines both types of information.

Neville et al. (2003) discuss this problem, and discuss a number of possible solutions. In the combined method they propose, the structure of the graph remains the same; the edges of the graph are given

weights that correspond to the similarity between the nodes they connect, then a graph clustering algorithm is applied to them. Neville et al. compare different graph clustering algorithms.

Witsenburg et al. propose an opposite direction: instead of introducing the content information in the graph (in the form of edge weights that indicate the similarity between nodes), they inject the structure information into the similarity function, after which a standard clustering algorithm is used. One could say that Neville et al. map the hybrid clustering task onto graph clustering, whereas Witsenburg et al. map it onto standard clustering.

In the following section we discuss three similarity measures that fit into Witsenburg's framework.

5 SIMILARITY MEASURES: CONTENT-BASED, STRUCTURE-BASED, HYBRID

Consider a graph $G(V, E, \lambda)$. We distiguish the following types of similarity measures.

Content-based measures: a content-based similarity measure $S : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}$ is one that compares two nodes by looking at their annotations (or "contents"), not taking their environment (the graph structure) into account.

Structure-based, or contextual, measures: a structure-based similarity measure is one that takes the context of the nodes into account, but not the contents of the nodes themselves (though it may be exploiting the contents of neighboring nodes, for instance). In the framework described above, this implies that a structure-based similarity measure $S' : \mathcal{A}' \times \mathcal{A}' \rightarrow \mathbb{R}$ implies that the context of a node can be mapped into a space $\mathcal{A}'$, which can be seen as an alternative annotation; this annotation contains all the structural information about the node that will be used to determine similarities; once the mapping has been made, each node is treated as an independent point in the input space. (In the terminology of propositional and relational learning, this process is called "propositionalisation").

Hybrid measures: these are measures that combine both information about the content of a node, and about the structure surrounding it.

In all three of the above cases, a similarity matrix M can be constructed, where M_{ij} indicates the similarity between node i and node j. From here on, we will denote with M the similarity matrix corresponding to the content-based similarity measure S, and with M' the similarity matrix corresponding to the structure-based measure S'. In this context we are assuming that there is a natural similarity measure defined over $\mathcal{A}$; what we are interested in, is ways to define M'.

M' will be using structural information, and this structural information is naturally encoded in the adjacency matrix A of the graph G. One way to define M' is the following equation:

$$M' = M \times A + A \times M \tag{1}$$

or, in terms of individual elements,

$$m'_{ij} = \sum_{n=0}^{N} (m_{in} \cdot a_{nj}) + \sum_{n=0}^{N} (a_{in} \cdot m_{nj}) \tag{2}$$

The practical meaning of the values in M' can easily be understood when taking a closer look at equation 2. Considering the first part ($\sum_{n=0}^{N}(m_{in} \cdot a_{nj})$) for every a_{nj} it holds that it is 0 when there is no relation between node n and node j and 1 otherwise. Thus, this first part will sum all values m_{in} for which a_{nj} is equal to 1. This can be described by saying that the first part gives the sum of all similarities of node i to all neighbours of j in the graph describing the relations. Analogously the second part is the sum of all similarities of node j to all neighbours of i.

In some cases, the above formula may have an unwanted side-effect. By summing similarities of neighbours, we get the effect that nodes with high degree will tend to score higher with respect to their similarity in S' than nodes with low degree. To counter this effect, we can use average values instead of sum; to that aim, equation 2 needs to be changed into equation 3.

$$m'_{ij} = \frac{1}{2} \cdot \left(\frac{\sum_{n=0}^{N}(m_{in} \cdot a_{nj})}{\sum_{n=0}^{N} a_{nj}} + \frac{\sum_{n=0}^{N}(a_{in} \cdot m_{nj})}{\sum_{n=0}^{N} a_{in}} \right) \tag{3}$$

The constant '$1/2$' in equation 3 ensures that all values in M' are in the same range as the values in M.

In the remainder, we will denote with M' the matrix defined according to equation 3.

We now have a similarity measure S that looks only at the content (annotations) of nodes to compare them, and a measure S' that looks at the content of neighboring nodes to compare two nodes, but never compares the content of the nodes directly. One could also combine the two into a hybrid form; a natural way of combining them is taking the average of the two:

$$M'' = (M + M')/2 \tag{4}$$

The definitions of M' and M'' are rather ad hoc; many other ways of defining similarity measures for nodes in a graph can be defined. We are not trying to argue here that these definitions are useful in all cases, or figure out under which conditions they are useful. The point that we are trying to make is that multiple definitions of similarity may make sense, and that depending on the task, one may be more suitable than the other. In fact, this effect may be visible even within a single dataset, as our experiments will show. The same experiments will also show that the similarity measure may influence the results of a learning procedure in some systematic way.

6 EXPERIMENTS

6.1 Cora Data Set

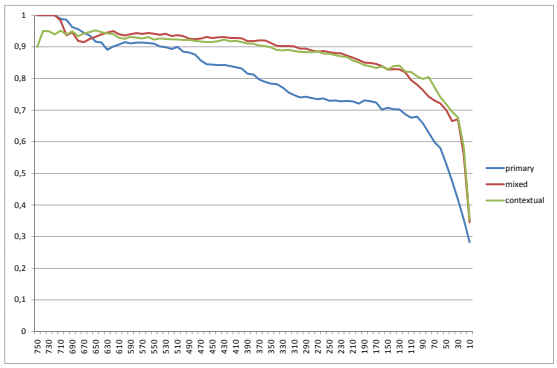
We have experimented with the well-known Cora data set [2]. This is a graph-structured data set containing 37,000 nodes; each node represents a scientific paper; two nodes are linked if they cite each other (the links are undirected: it is not known which is the citing and cited paper); each node is annotated with the abstract of the paper it represents (more precisely: with a bag-of-words representation of this abstract), and with a label denoting one or more subject classes (from 70 possible classes).

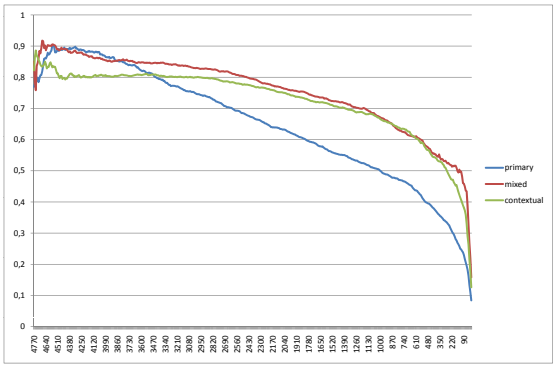
The content-based similarity S is defined over the bag-of-words only, not over the class labels. The matrix elements of M are computed as follows:

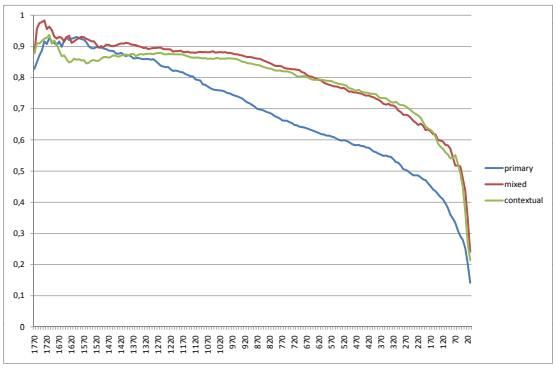
$$m_{ij} = \frac{1}{2} \cdot \left(\frac{|b_i \cap b_j|}{|b_i|} + \frac{|b_i \cap b_j|}{|b_j|} \right) \tag{5}$$

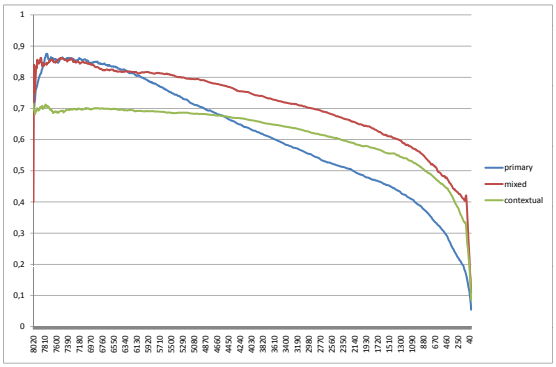
with b_i the bag of words corresponding to node i. It can easily be seen that this is the average ratio of words that are in common between two papers. The adjacency matrix A was created by taking into account the citation relation: a_{ij} is 1 when paper i cites paper j or is cited by it, and 0 otherwise.

Five subsets of the Cora dataset were created. Each subset is clustered with agglomerative hierarchical clustering: once with the content based (primary) similarity, once with the contextual (secundary)









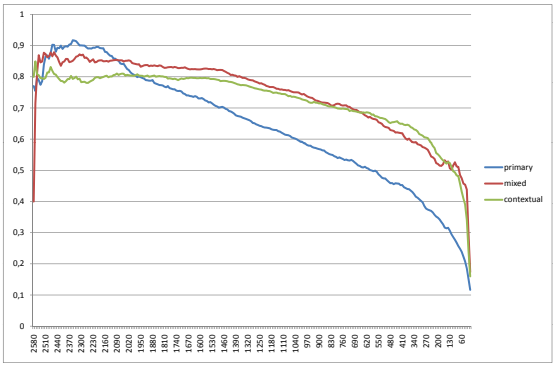


Figure 2. Evolution of clustering quality in terms of number of clusters. While content-based similarity yields better (more coherent) clusters when many small clusters are produced, contextual similarity performs better when fewer large clusters are produced. The hybrid similarity measure tends to globally outperform or coincide with the best of these two.

similarity and once with the hybrid (mixed) similarity. On the x-axis is the amount of clusters, which goes from $|V|$ to 1 following the working method of agglomerative hierarchical clustering.

Every ten iterations the quality of the clustering is calculated. To do so, only clusters with size greater than one are considered. For these clusters the quality is calculated by taking the average Jaccard index of the classes of each pair of elements in this cluster. More specifically, if we have two nodes i and j, each of which are labelled with a set of classes c_i and c_j, their Jaccard index is $|c_i \cap c_j|/|c_i \cup c_j|$. The quality of a clustering is the average Jaccard index between any two nodes in the same cluster, averaged over all clusters.

The graphs clearly show some remarkable effects for which we do not have a straightforward explanation, but the existence of which we find interesting. Starting with many small clusters, the contextual similarity measure performs less well than the content-based similarity measure (that is, the clusters it forms are less coherent with respect to the classes of their elements). As the number of clusters goes down and the clusters become larger, however, there is a point where the contextual similarity starts performing better than the content based similarity.

Another remarkable fact is that the hybrid similarity performs well globally: it appears quite successful at combining the advantages of both other similarity measures.

7 CONCLUSIONS

In this paper we have argued that, while it is generally accepted that the input representation influences the quality of the learning process, it may be better to study input representation in terms of the similarity measure that is defined over the input space, rather than in terms of the features that are being used. In the context of learning from graphs (but also in other contexts), it may be possible to define such a similarity measure without specifying precisely the features on which this similarity measure is based. With some experiments on the Cora dataset, we have shown that the similarity measure that one uses indeed influences the results, and moreover, in this case it appears to influence them in a systematic way. This suggests that the influence of similarity measures on learning performance can, and should, be studied. Up till now, little attention has been paid to this particular aspect of learning bias. It may be useful to increase that attention in the future.

ACKNOWLEDGEMENTS

This research is funded by the Dutch Science Foundation (NWO) through a VIDI grant.

REFERENCES

[1] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, 'CRISP-DM 1.0: Step-by-step data mining guide', Technical report, CRISP-DM Consortium, (2000).

[2] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, 'Automating the construction of internet portals with machine learning', *Information Retrieval*, **3**, 127–163, (2000).

[3] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[4] J. Neville, M. Adler, and D. Jensen, 'Clustering relational data using attribute and link information', in *Proceedings of the Text Mining and Link Analysis Workshop, 18th International Joint Conference on Artificial Intelligence*, (2003).

[5] T. Witsenburg and H. Blockeel, 'A method to extend existing document clustering procedures in order to include relational information', in *Proceedings of the 6th International Workshop on Mining and Learning with Graphs*, pp. 1–3, (2008).